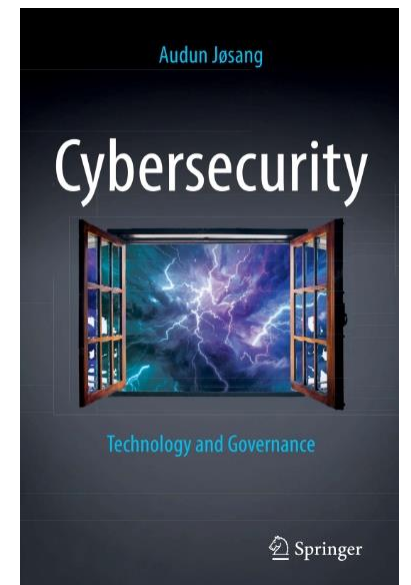


Chapter 5: Key Management and PKI

Cybersecurity: Technology and Governance

Audun Jøsang,

1st Edition, Springer Nature, 2024.



Overview

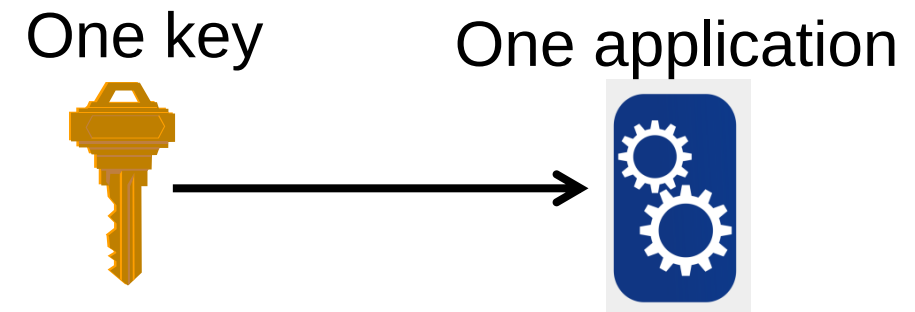
- Key Management
 - Why key management
 - Crypto periods
 - Recommended key sizes
 - Key Management Process
- PKI
 - The challenge of key distribution
 - Importance of authenticity for public keys
 - X-509 certificates, generation and validation
 - Trust models
 - Internet PKI



The Importance of Key Management for Cryptography

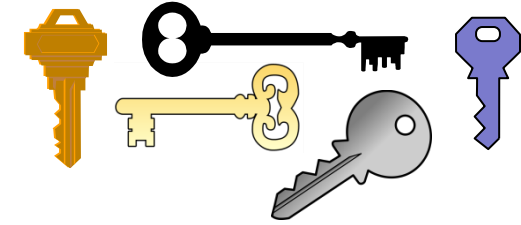
- The strength of cryptographic systems depends on:
 - The size of the key
 - The Strength of Cryptographic Algorithms/Protocols
 - The correctness and integrity of the HW/SW implementation of cryptographic algorithms/protocols
 - Key Management
- Key management provides the foundation for the secure generation, storage, distribution and destruction of keys.
- Poor key management can easily lead to breaches of the security of systems that are based on cryptography.

One application per key



- A single key should only be used for a single application
 - e.g. encryption, authentication, key encapsulation, random number generation or for digital signature
- Using the same wrench for multiple applications may impair the safety of one or more applications.
- Restricting the use of a key limits the damage that can occur if the key is compromised.
- Follows the same principle that passwords should not be reused between different services.
 - Reusing passwords can make many services insecure if the password is leaked or cracked on one of the services.

Types of cryptographic keys



- Crypto keys are categorized according to:
 - whether they are public, private or symmetric (secret)
 - Intended use
 - whether they are static (long life) or volatile (short life)
- 19 different types of cryptographic keys defined in : NIST Special Publication 800-57, Part 1, "Recommendation for Key Management"

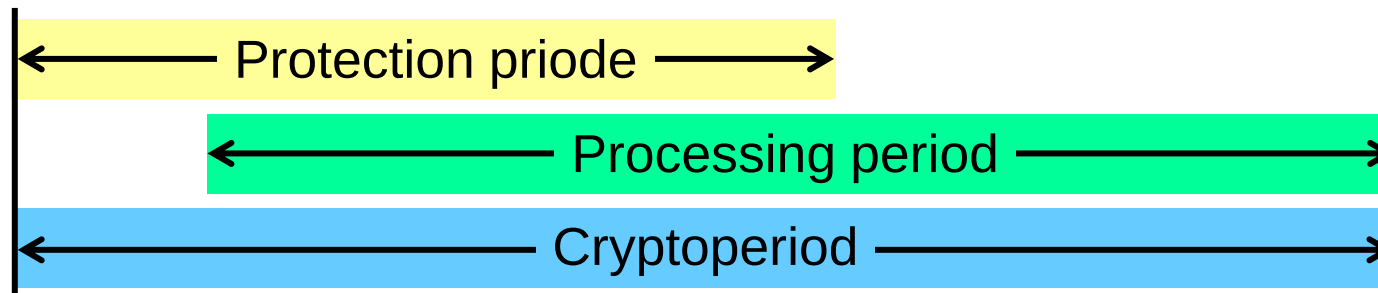
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf>

Kryptoperiode



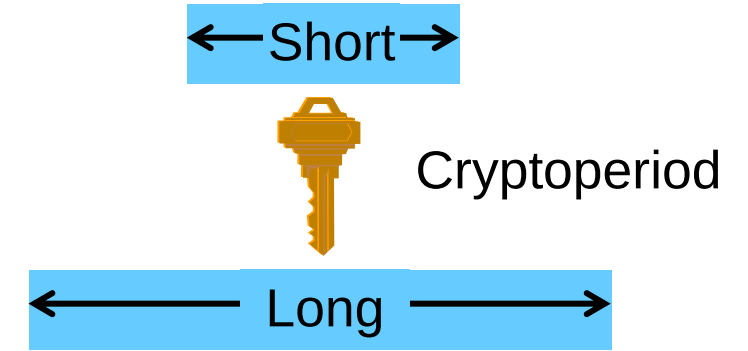
- The cryptoperiod is the time period of authorized use of a particular key
- The cryptoperiod is important because it:
 - Limits the amount of data protected by a given key, assuming that the key can potentially be cryptanalyzed.
 - Limits the amount of exposure and damage, if a single key is compromised.
 - Limits the use of a particular algorithm to the estimated secure lifetime.

Cryptoperiods



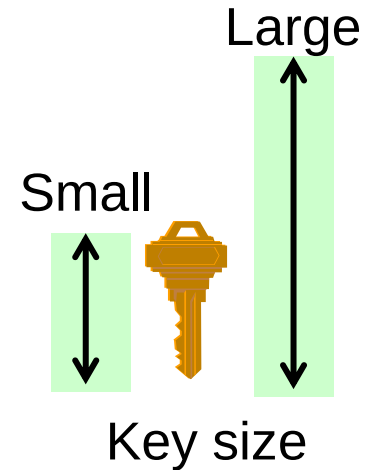
- A key can be used for protection and/or processing.
- Protection period:
 - The key is used to encrypt (secret symmetric or public asymmetric key)
 - The key is used to generate a digital signature (private asymmetric key)
- Processing period:
 - The key is used to decrypt (secret symmetric or private asymmetric key)
 - The key is used to validate a digital signature (public asymmetric key)
- The cryptoperiod lasts from the beginning of the protection period to the end of the processing period
 - The processing period can continue after the end of the protection period

Factors that determine the cryptoperiod



- In general, as the sensitivity of the information or application increases, the crypto period should be reduced to limit potential damage in the event of compromise.
- Short cryptoperiods can however be counterproductive, especially where availability of data/applications is important, and there is significant overhead and/or some probability of failure in key generation or the key distribution process.
-
- The length of the cryptoperiod is therefore a trade-off

Recommended sizes (in bits) of symmetric (secret) keys



- Symmetric secret keys are used in block ciphers such as AES.
- The size of the key determines the average time required for cryptanalysis with exhaustive search, i.e., trying all keys.
- Less and less time for exhaustive search due to ever-increasing computing power.
- Owners of systems must set the key size based on requirements for how many years a crypto system should remain secure.

Recommended sizes for symmetrical keys

Ref: NIST SP 800-57

Security Strength		Through 2030	2031 and Beyond
< 112	Applying protection	Disallowed	
	Processing	Legacy use	
112	Applying protection	Acceptable	Disallowed
	Processing		Legacy use
128	Applying protection and processing information that is already protected	Acceptable	Acceptable
192		Acceptable	Acceptable
256		Acceptable	Acceptable

Equivalent sizes for asymmetric keys

Ref: NIST SP 800-57

Finite Field
Cryptography

Integer Factorization
Cryptography

Elliptic Curve
Cryptography

Security Strength	Symmetric Key Algorithms	FFC (DSA, DH, MQV)	IFC* (RSA)	ECC* (ECDSA, EdDSA, DH, MQV)
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256-383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384-511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

* The security-strength estimates will be significantly affected when quantum computing becomes a practical consideration.

Quantum computers' potential threat to cryptography

- NIST (US National Institute of Standards and Technology) suggests the possibility of powerful quantum computers in the late 2020s
- Consequence for traditional asymmetric crypto:
 - ~~– RSA~~
 - ~~– Elliptical Curve Cryptography (ECDSA)~~
 - ~~– Finite Field Cryptography (DSA)~~
 - ~~– Diffie-Hellman key exchange~~

➤ Need for new post-quantum algorithms
- Consequence for symmetric crypto:
 - AES

➤ Double key size (256 bits) is OK
- Consequence for hash functions:
 - SHA-2 and SHA-3 with 512-bits hash

➤ Double hash size (512 bits) is OK

Key Generation

- Most sensitive of all cryptographic operations.
- Key generators in software or hardware must be protected to prevent:
 - leakage, weakening or forgery of keys;
 - leakage, attenuation, or falsification of IVs (initialization vectors).
- Keys must be randomly selected from the entire key compartment
 - e.g. 128 bit key provides a key space of 2^{128} different keys
 - Each key should be equally likely

When keys aren't random

- Edward Snowden revealed in 2013 that RSA Security Inc. (prominent security company) was bribed by the NSA with \$10 million to implement a weak random number generator in its BSAFE security products.
- The NSA was able to predict random numbers and regenerate the same secret keys as those used by RSA's customers.
- With the secret keys, the NSA could read all data encrypted with RSA's BSAFE security product.



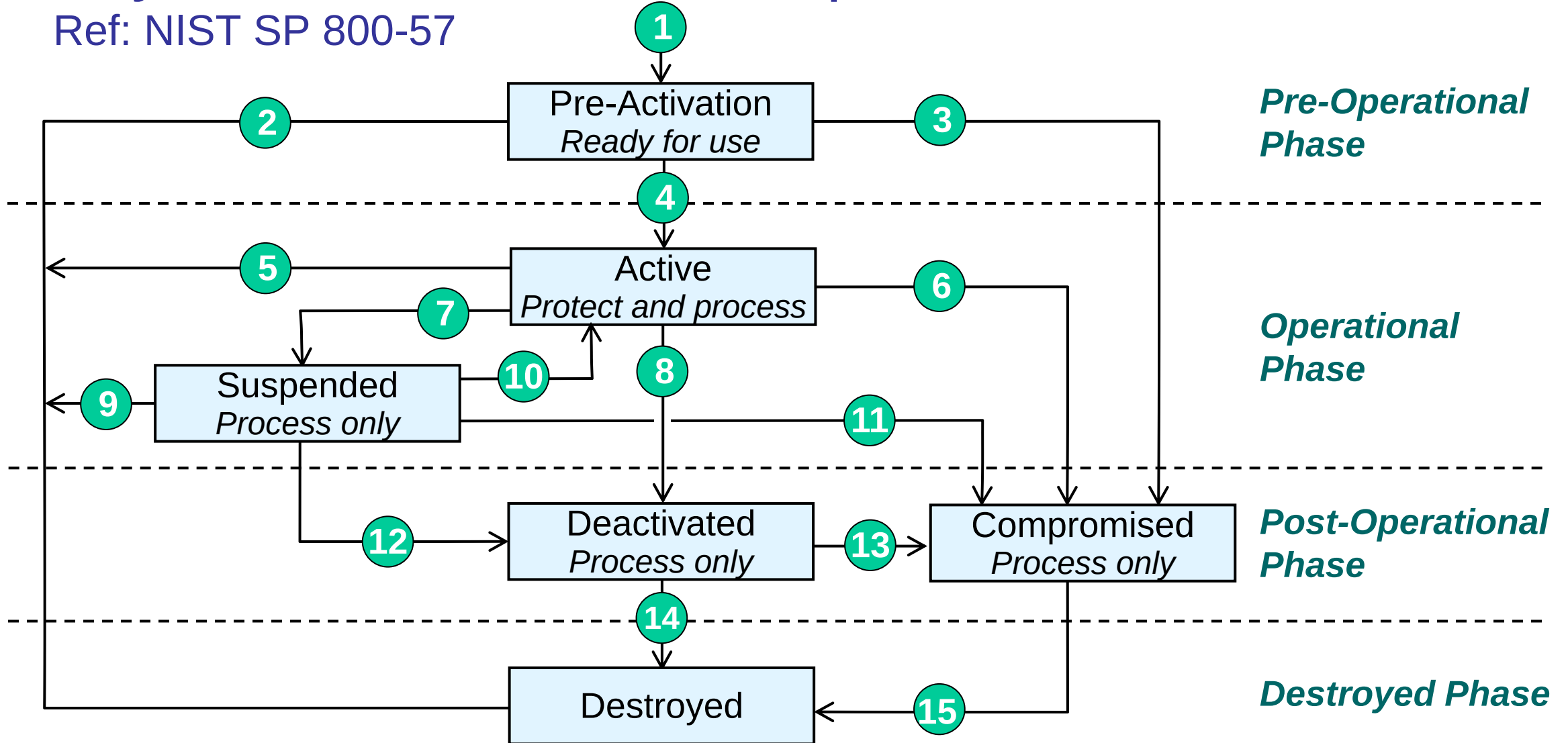
Edward Snowden

Compromise of cryptographic keys

- Key compromise is if it is known or suspected that an unauthorized entity has obtained a secret/private key.
- When a secret/private key is compromised, the use of the key for protection must cease immediately, and the key is defined as compromised.
- A compromised symmetric key, or the public key corresponding to a compromised private key can be used for continued processing (decryption) of encrypted data, or signature verification.
- All users of the key must be informed and made aware of the risks.
- Continued key use depends on the risk and on your organization's key management policies.
- Key compromise is most serious is when it happens without its detection.

Key states, transitions and phases

Ref: NIST SP 800-57



Protecting keys

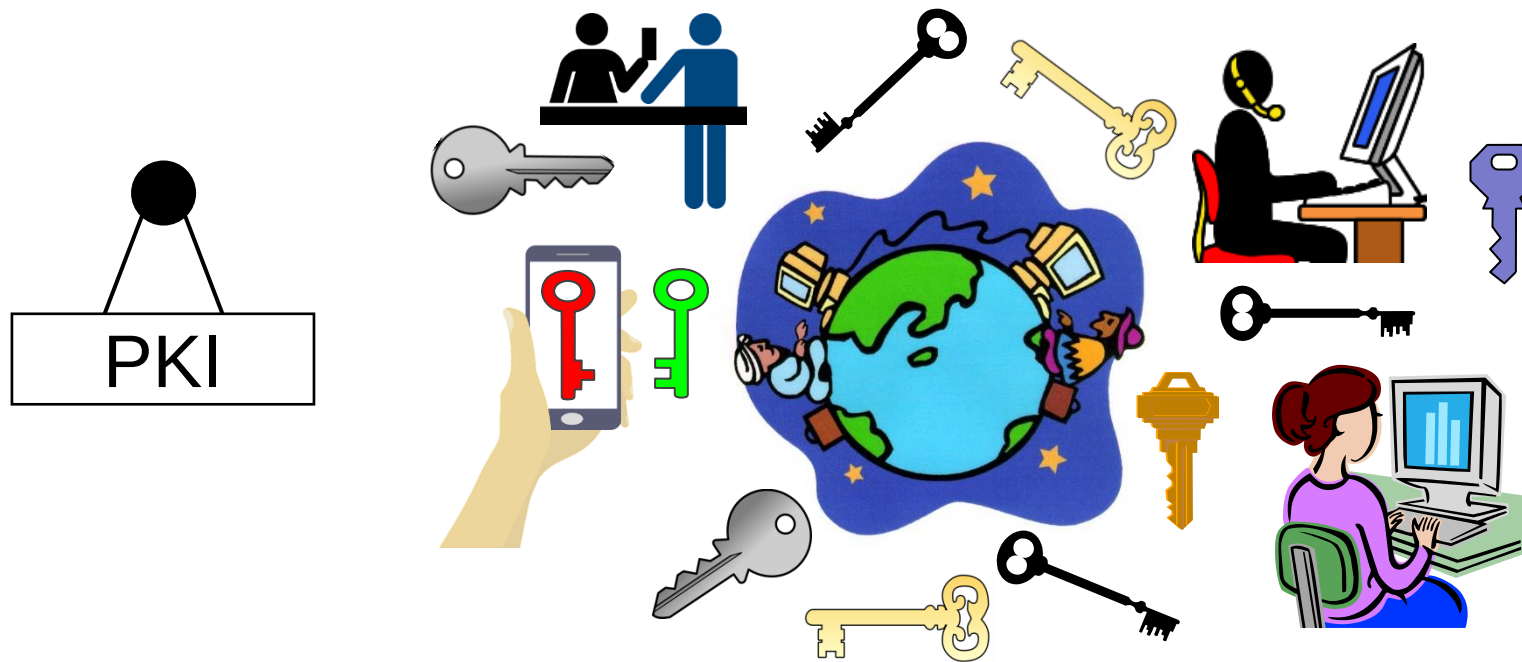
- Secret keys for symmetric ciphers
 - Never saved or transferred in clear
 - Can use hierarchy: session keys encrypted with master key
 - Master key protection:
 - Locks and guards
 - Tamper-proof devices
 - Password protection
- Keys for asymmetric ciphers
 - Private keys need confidentiality protection (as for secret keys)
 - Public keys need integrity/authenticity protection (see "PKI" next)

Key destruction

- Keys must not exist in volatile memory or on permanent storage media after destruction
 - Methods for destroying keys are e.g.
 - Easy delete operation on system
 - Risk that the key is still present in the Recycle Bin or on disk sectors
 - Special delete operation on computer
 - such as overwriting memory and disk so that it leaves no residual data,
- Degaussing of magnetic storage media with a strong magnetic field
- Destruction of physical device e.g. by crushing or exposing to high temperature
- Destroying the master key will logically also destroy child keys encrypted under the master key

PKI (Public-Key Infrastructure)

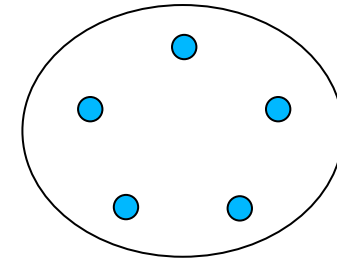
Cryptography solves security challenges in open computer networks, but creates challenges for key distribution.



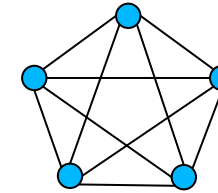
Asymmetric cryptography simplifies key distribution but requires PKI which creates challenges for trust management.

Key distribution challenge

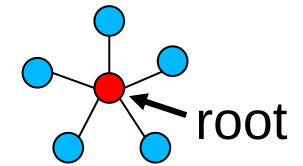
- Assume a computer network (e.g., Internet) with n nodes
- Each pair of nodes needs a separate key to communicate securely with cryptographic protection
- How many secure key distributions are needed?
 - Symmetrical secret keys: distribution requires **confidentiality**.
 $n(n-1)/2$ distributions, quadratic growth.
Impractical in open networks.
 - Asymmetric public keys: distribution requires **authenticity**.
 $n(n-1)$ distributions of public keys, quadratic growth
Impractical in open networks.
 - Asymmetric public keys with PKI: distribution requires **authenticity**.
 - 1 root key is distributed to all n nodes
 - Linear growth
 - ... much easier, yet relatively challenging



Nodes of a
computer
network



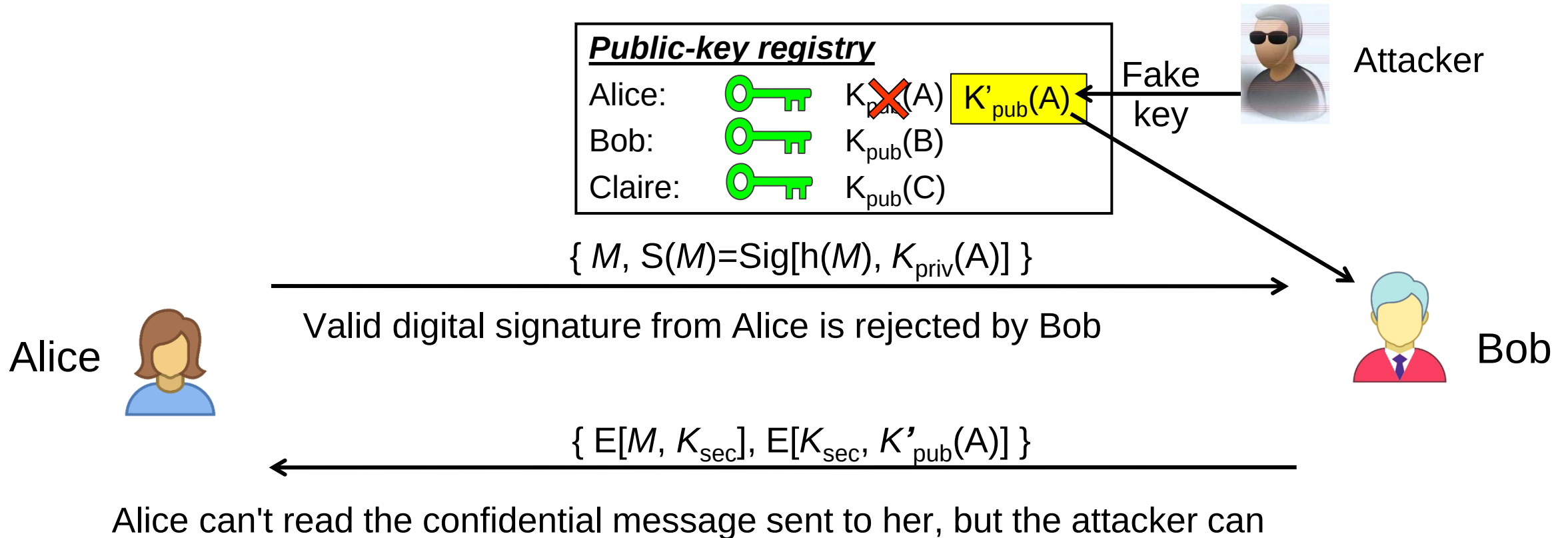
n nodes
 $n(n-1)/2$ edges



n nodes
 n edges

The problem of forgery of public keys

- Suppose public keys are stored in a public registry
- What is the consequence if the attacker forges Alice's public key in the registry?



PKI: Infrastructure for secure distribution of public keys

- To protect authenticity, public keys must be digitally signed and distributed as "public key certificates".
- The purpose of a PKI is to guarantee authenticity of public keys and simplify key distribution. PKI consists of:
 - **Policies** (to define the rules for managing certificates)
 - **Technologies** (to generate, deploy, store, and validate certificates)
 - **Procedures** (related to the management of certificates)
 - **Trust model** (how the certificates are cryptographically linked to each other)

X.509 Public Key Certificates

- The format for public-key certificates is defined in the X.509 standard
- A public key certificate is a record of data, including the subject's (domain) name and its public key, all digitally signed by a CA (Certificate Authority).
- Creates a logical binding between the (domain) name and its public key
- An authentic copy of the CA's public key is required to validate the certificate, i.e. verify that the digital signature is correct.

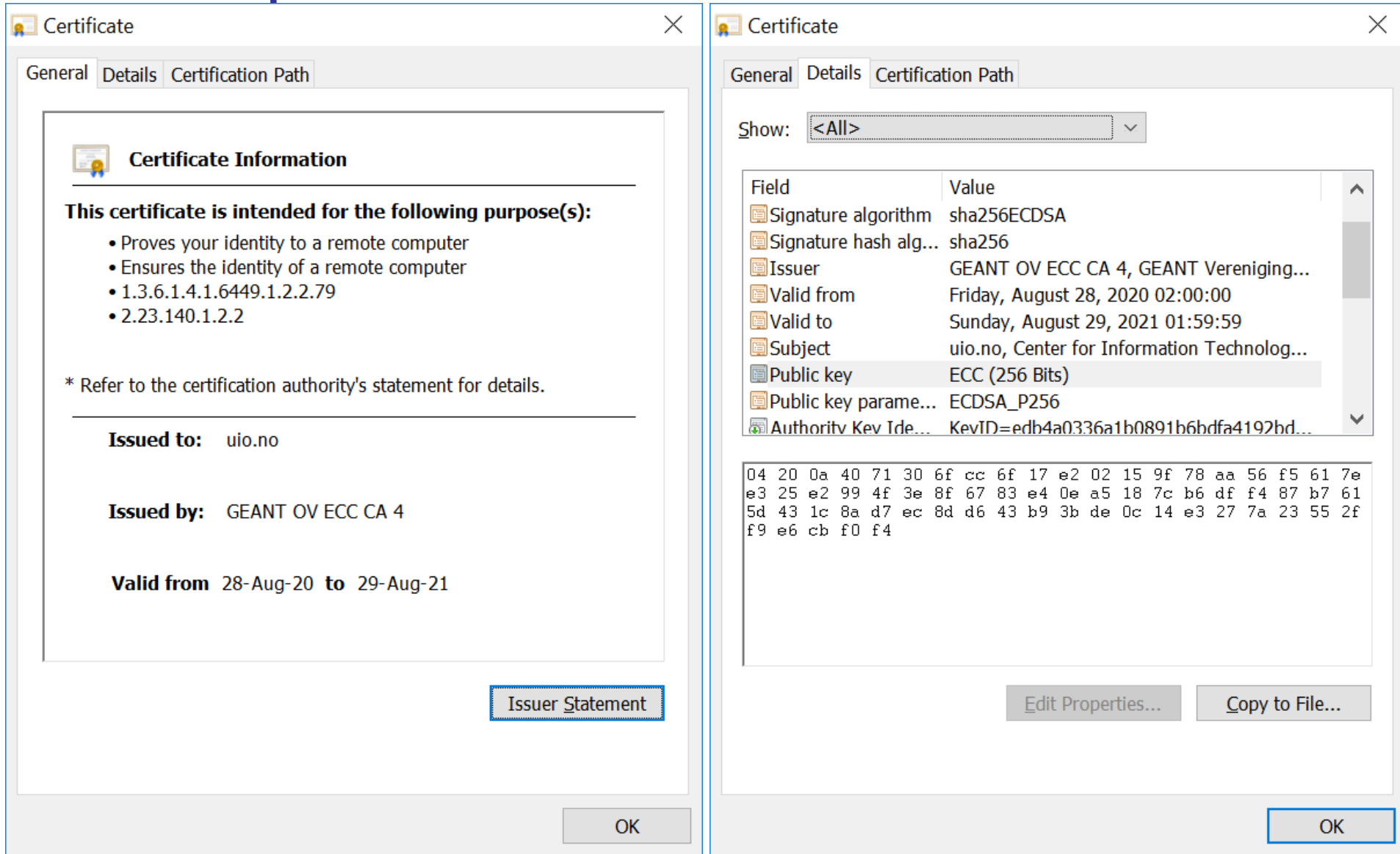
X.509 Digital Certificate

- Version
- Serial Number
- Algorithm Identifier
- Issuer CA
 - Name
- Subject key owner

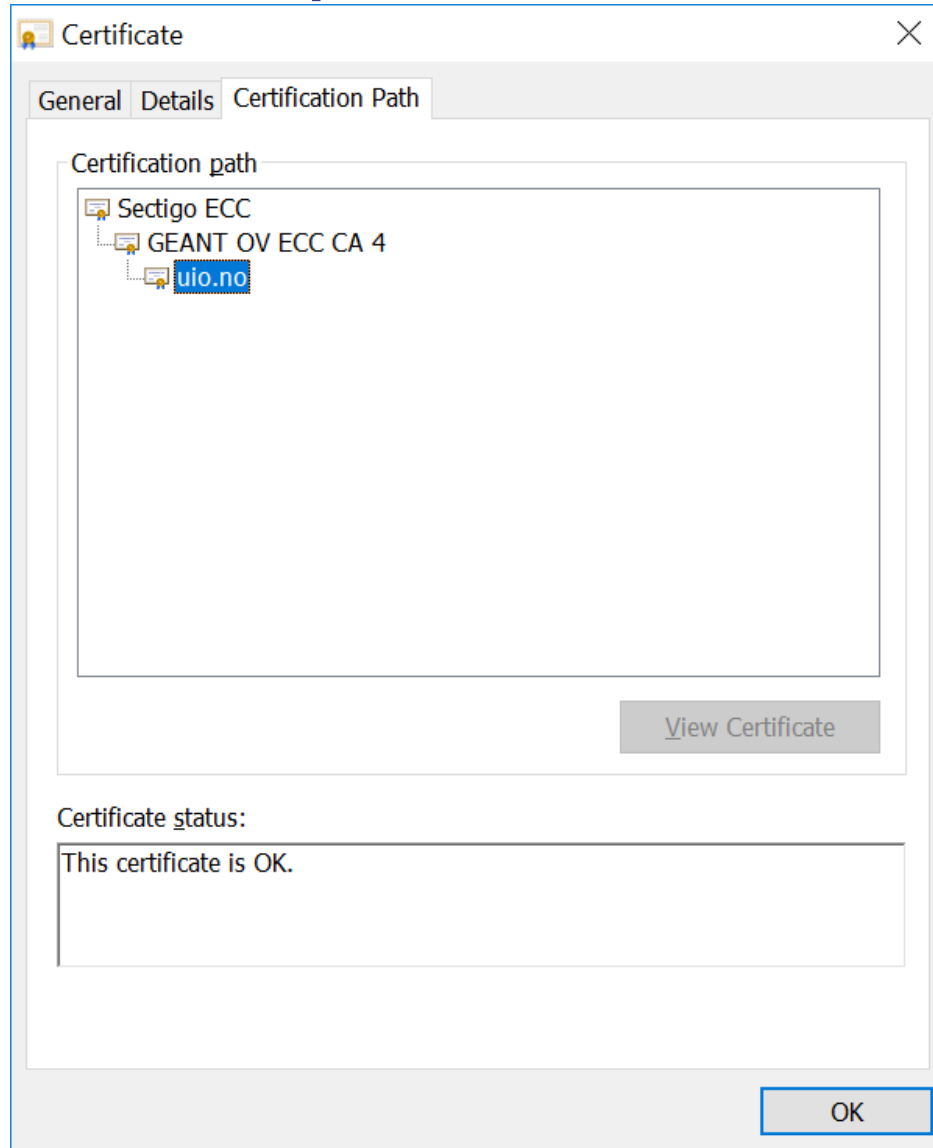
- Name
- Public Key 

- Validity Period
- Extensions
- Digital Signature 

Example of X.509 certificate



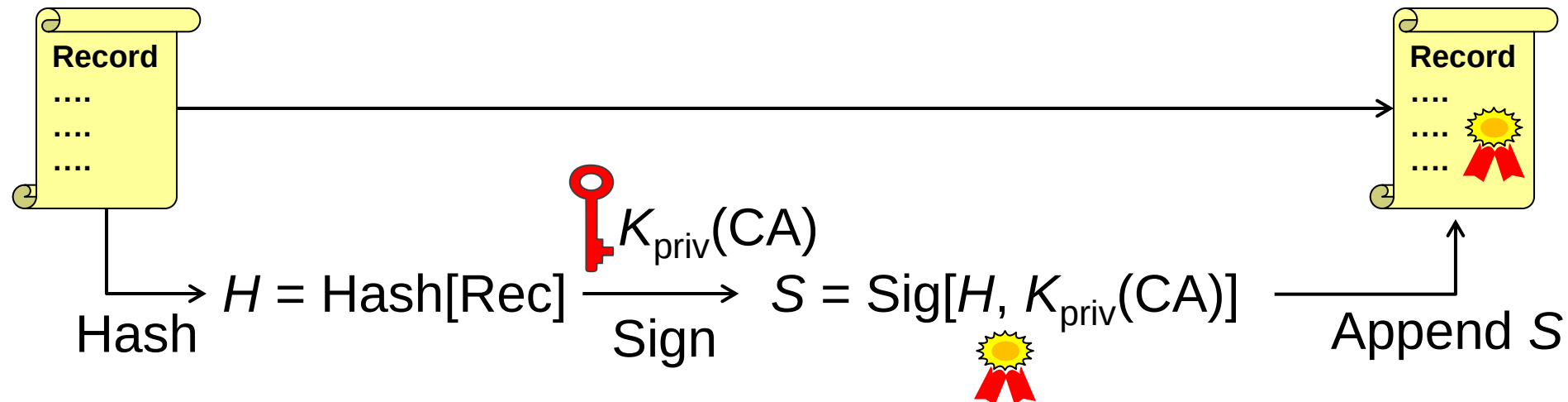
Example of X.509 certificate (cont.)



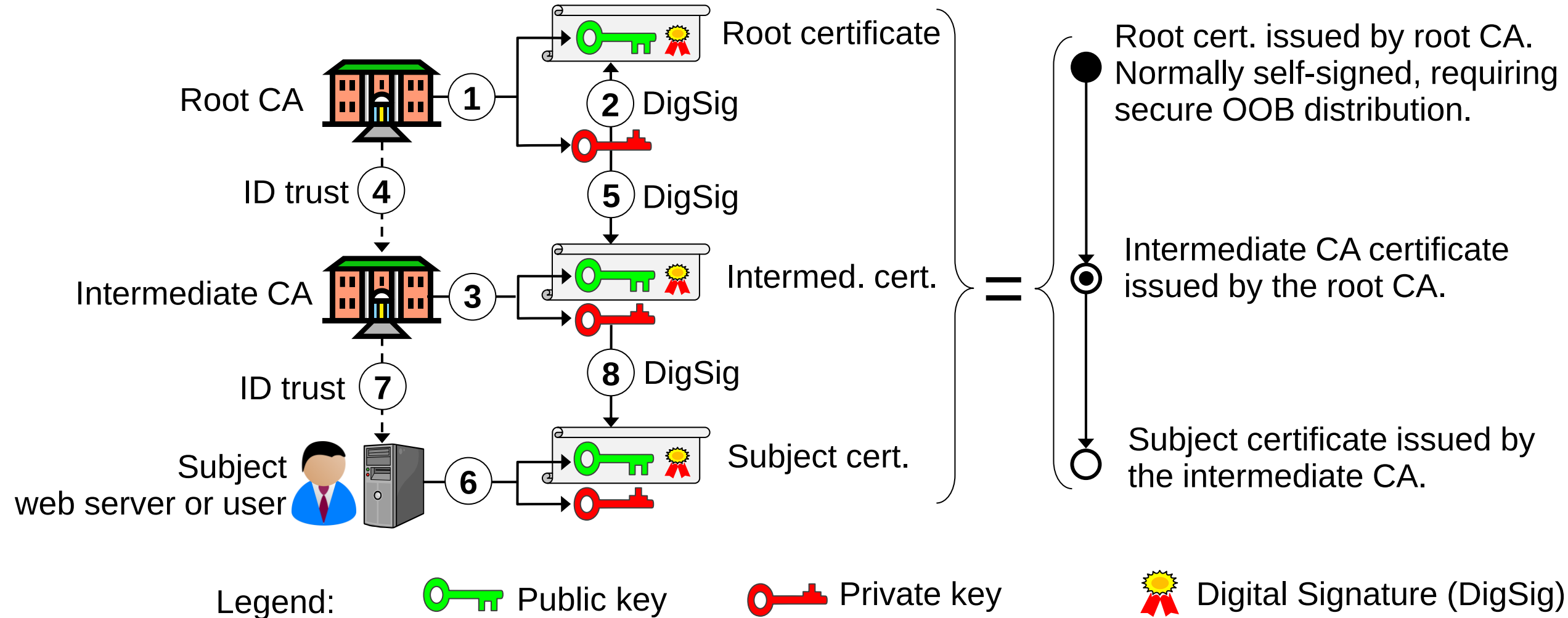
- The certificate path is the chain of X.509 certificates from the root of the subject's certificate
- The root certificate is pre-stored as part of the browser
- Each browser stores a few hundred root certificates.
- Intermediate certificates can be pre-stored or downloaded in real-time.
- Validation of subject certificates starts from the root.

How to generate a public-key certificate

1. Collect all the data fields, including the subject's (domain) name and public key, in a data record Rec
2. Hash the data record
3. Sign the hash value
4. Append the digital signature to the data record



Chaining certificates

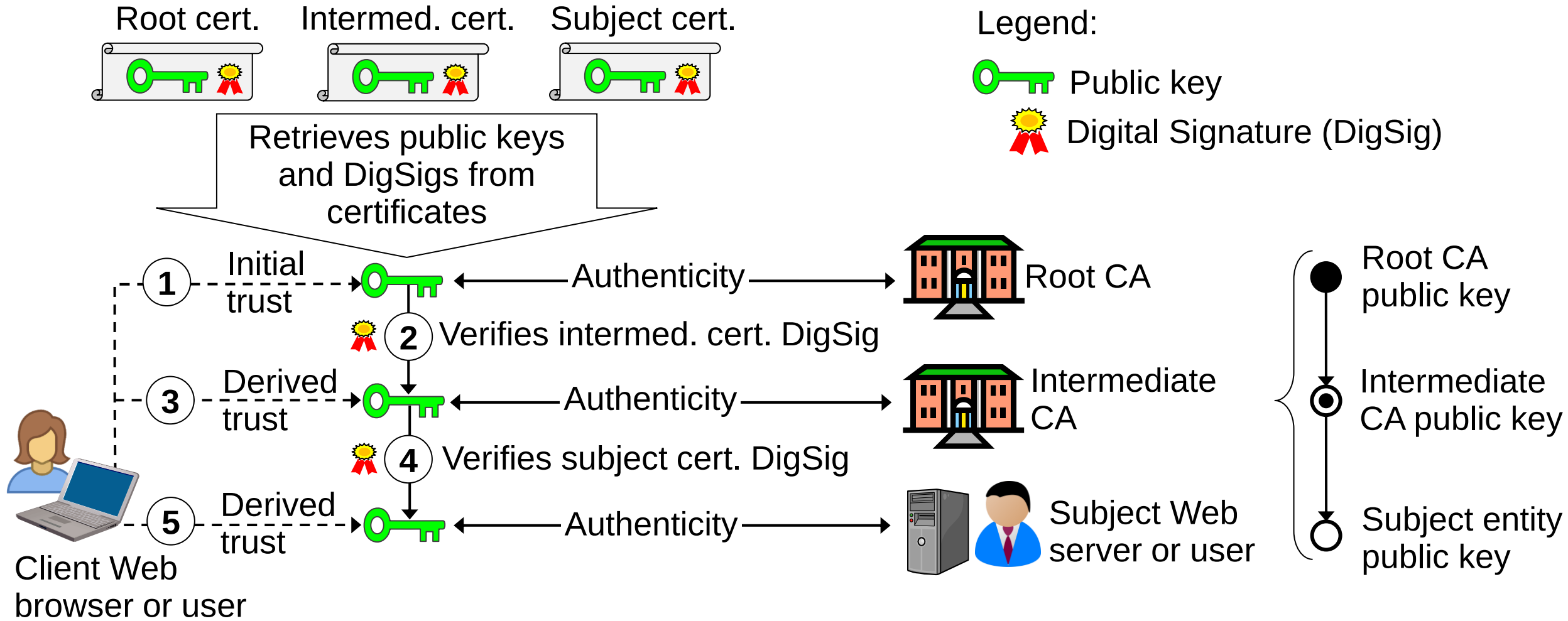


Why self-signed root certificates?

- Some people mistakenly believe that a root certificate is authentic just because it is self-signed.
- The term "self-signed" can be misleading
 - Can give a false impression of security
 - Can be used to hide fake certificates
 - Can be used to spoof server certificates in TLS stripping attacks
- Self-signing provides absolutely no security
- Possible useful purposes of self-signing:
 - Provides a checksum to detect unintentional corruption
 - X.509 certificates have a digital signature field, so a blank field can cause applications to malfunction. A self-signature is a way to fill in the blank



Validation of certificates



Trust models

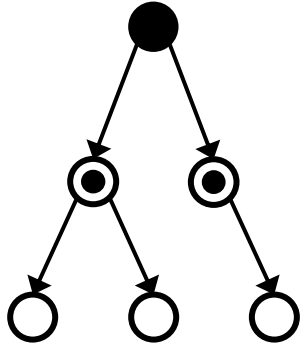
Legend:

● Self-signed root certificate

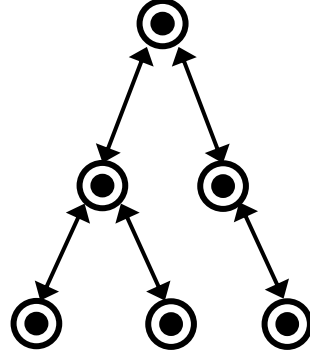
⊙ CA-signed intermediate CA certificate

○ CA-signed subject certificate

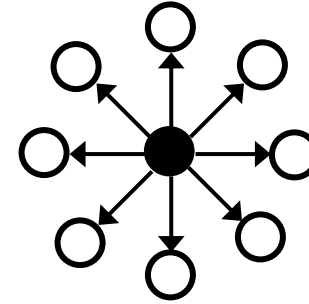
● Self-signed subject certificate



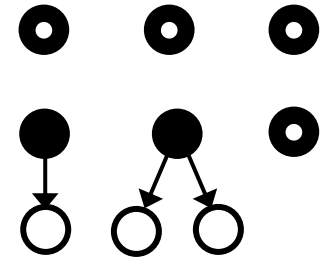
Simple tree hierarchy



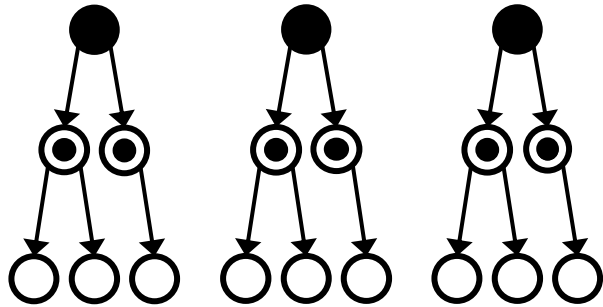
Bi-directional hierarchy



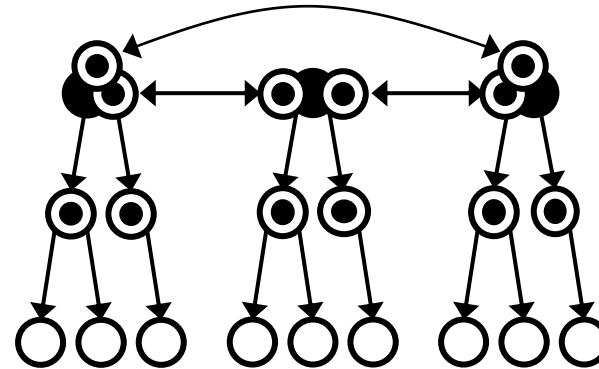
User-centric hierarchy
(Subject user is own CA)



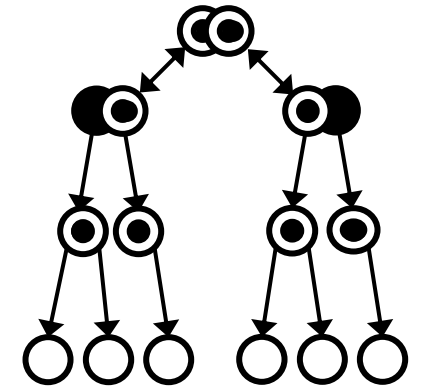
Unstructured PKI



Multiple isolated tree hierarchies
(Internet PKI)

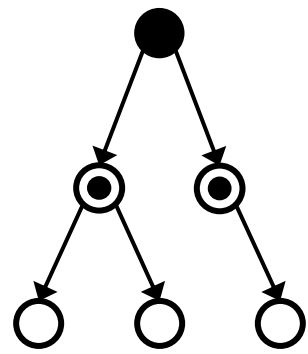


Cross-certified simple hierarchies
(Mesh PKI)



PKIs with a bridge CA

Simple hierarchy



Legend:

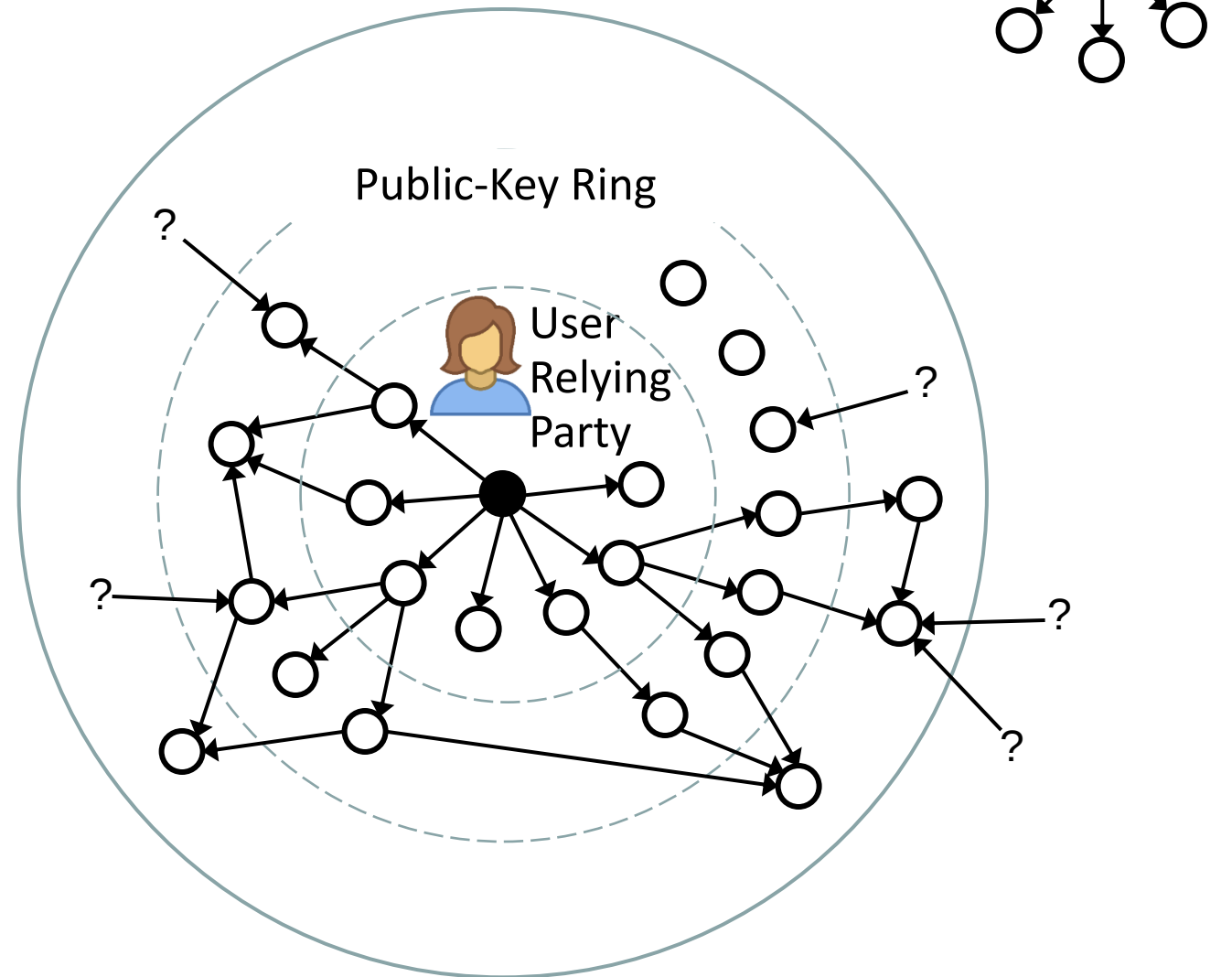
- Self-signed root certificate
- ⊙ CA-signed intermediate CA certificate
- CA-signed subject certificate

- Advantages:
 - Works well in highly structured organizations such as military networks
 - Simple trust structure
 - Works well in closed/isolated data networks
- Disadvantages:
 - All subject entities must trust the same root CA
 - Root-CA compromise leads to total security breakdown
 - Doesn't scale to open data meshes

User-centric PKI

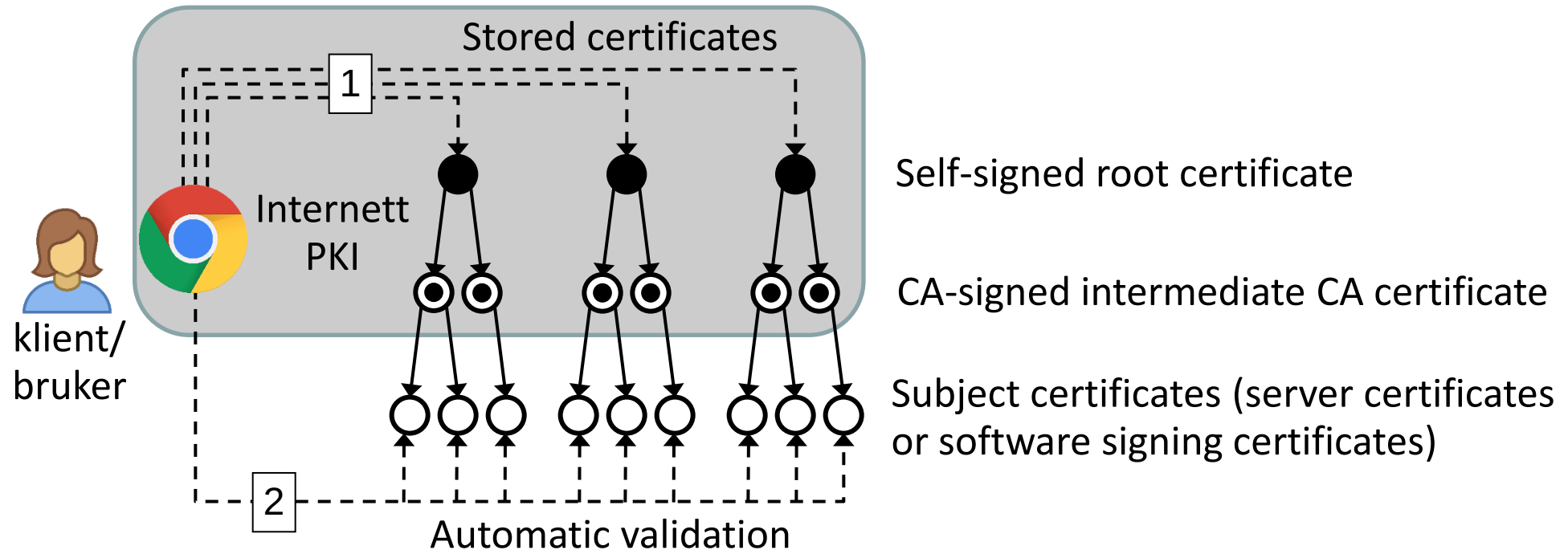
Each user is their own root CA

- Each entity signs public keys to others when they are verified to be authentic.
- Each user has their own "public keychain"
- Public keys signed by other trusted people can also be considered authentic.
- This model is used in PGP (also called GPG)



Internet PKI: Silo-hierarkies

(PKI used by browsers for https)

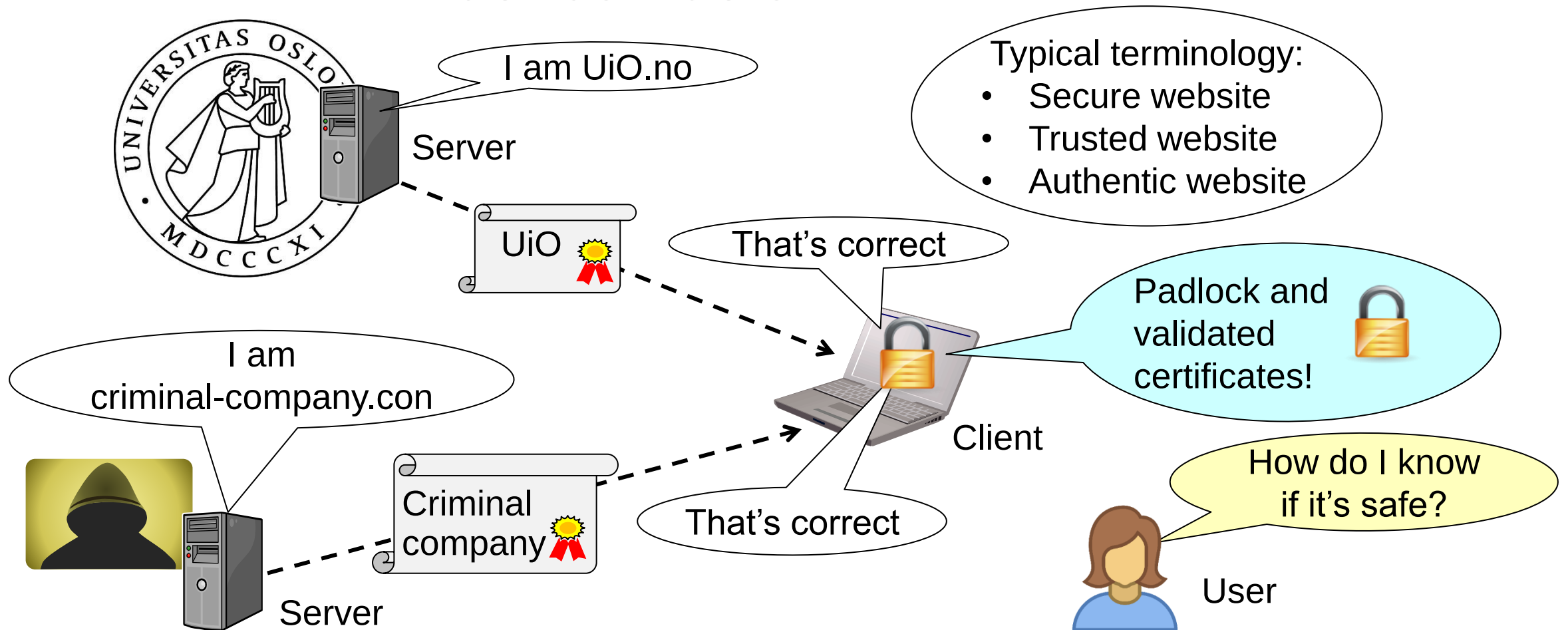


The Internet PKI consists of a few hundred silo hierarchies, where the (root) CA certificate for each hierarchy is installed as part of the browser. Each PKI is its own silo, with no connection to other silo PKIs. New/updated root certificates can be imported by the user or through (automatic) software updates

Internet PKI and Fake Certificates

- Certificates are automatically validated by the browser checking the digital signature, and there is a match between the certificate's domain name and the website's domain name
- Criminals can purchase legitimate certificates that are automatically validated by Web browsers
- Legitimate certificates can be used in conjunction with phishing attacks, e.g. to create a fake website for a bank
- Fake websites may have legitimate certificates !!
- Server certificate validation is only syntactic authentication, not semantic verification of the website's authenticity
- Users who do not know the server's domain name cannot know beforehand whether it is fake or not

Trust confusion



- Criminal company is both authentic and unreliable/fake
- To avoid this confusion, we need to be specific about what we mean by trust
- X.509 certificates only guarantee authenticity, not reliability!

PKI Summary

- Encryption with public keys needs a PKI to be convenient
-
- PKIs are complex and expensive to operate
-
- Internet PKI is the most widely used PKI thanks to the distribution of root certificates with web browsers
-
- The security of the PKI depends on the integrity of the CAs
-
- PKI services are called "trust services" in the EU's Digital Agenda
-
- PKI and trust services form the basis for e-ID and e-government in the EU

End of presentation

